Scaling properties of a parallel implementation of the multicanonical algorithm

Johannes Zierenberg, Martin Marenz, Wolfhard Janke

Institute for Theoretical Physics
University of Leipzig

30.11.2012

BuildMoNa

Gefördert aus Mitteln der Europäischen Union

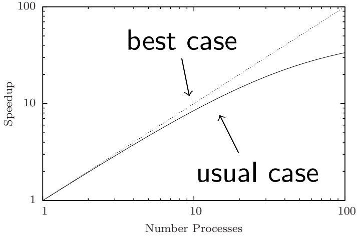
Europa fördert Sachsen. ESF Europäischer Sozialfonds

Outline

Why?

- Can a multicanonical simulation be parallelized in order to reduce my waiting time?

Why?

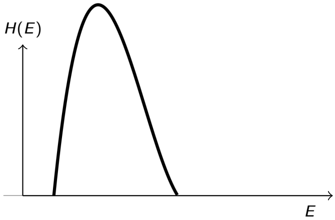


- Can a multicanonical simulation be parallelized in order to reduce my waiting time?
- If it can, where is the limit of cost-benefit?

Multicanonical Method[1]

The canonical partition sum can be rewritten

$$\mathcal{Z}_{\rm can} = \sum_{\{x_i\}} e^{-\beta E_i}$$

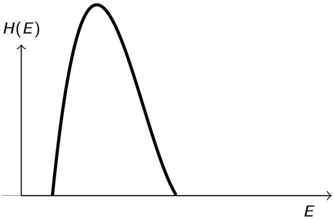


[1] B. A. Berg and T. Neuhaus, *Phys. Lett. B* **267** 249 (1991), *Phys. Rev. Lett.* **68** 9 (1992)

Multicanonical Method[1]

The canonical partition sum can be rewritten

$$\mathcal{Z}_{\mathrm{can}} = \sum_{\{x_i\}} e^{-\beta E_i} \to \mathcal{Z}_{\mathrm{muca}} = \sum_{\{x_i\}} W(E_i)$$



[1] B. A. Berg and T. Neuhaus, *Phys. Lett. B* **267** 249 (1991), *Phys. Rev. Lett.* **68** 9 (1992)

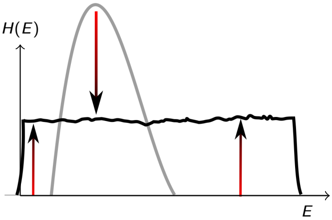
Multicanonical Method[1]

The canonical partition sum can be rewritten

$$\mathcal{Z}_{\mathrm{can}} = \sum_{\{x_i\}} e^{-\beta E_i} \rightarrow \mathcal{Z}_{\mathrm{muca}} = \sum_{\{x_i\}} W(E_i)$$

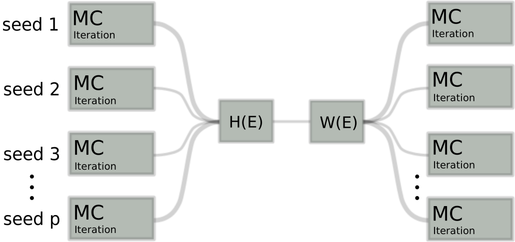
$$W^{(\mathrm{n}+1)}(E) = \frac{W^{(\mathrm{n})}(E)}{H^{(\mathrm{n})}(E)}, \text{ or}$$

$$W^{(\mathrm{n}+1)}(E) = F\left[W^{(\mathrm{n})}(E), W^{(\mathrm{n})}(E+\Delta E)\right]$$



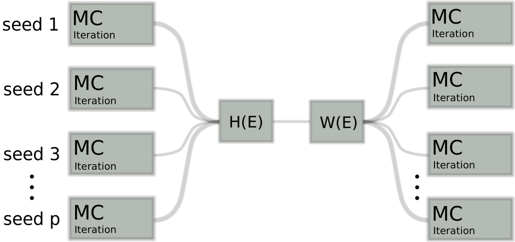
[1] B. A. Berg and T. Neuhaus, *Phys. Lett. B* **267** 249 (1991), *Phys. Rev. Lett.* **68** 9 (1992)

Parallel MUCA



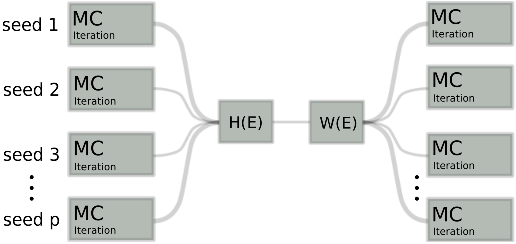
$$\sum_i H_i^{(n)}(E)$$

Parallel MUCA



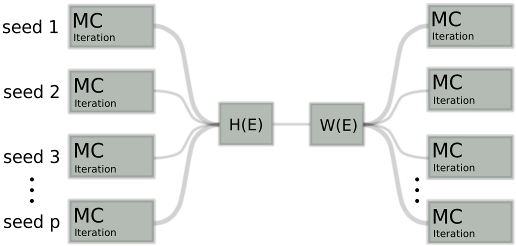
$$\sum_i H_i^{(n)}(E) = H^{(n)}(E)$$

Parallel MUCA



$$\sum_i H_i^{(n)}(E) = H^{(n)}(E) \rightarrow W^{(n+1)}(E)$$

Parallel MUCA



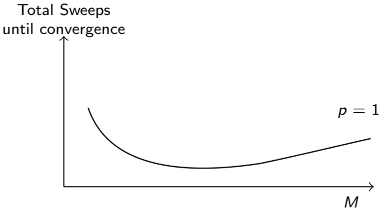
$$\sum_i H_i^{(n)}(E) = H^{(n)}(E) \rightarrow W^{(n+1)}(E) = W_i^{(n+1)}(E)$$

How to judge the speedup?

When changing the number of processes, the parallel implementation yields different simulations.

p : number of processes

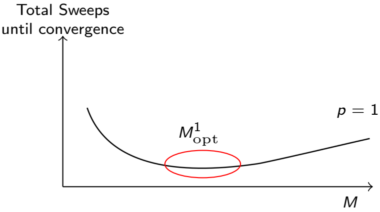
M : number of sweeps per iteration (for each process)



How to judge the speedup?

When changing the number of processes, the parallel implementation yields <u>different</u> simulations.

- p : number of processes
- M : number of sweeps per iteration (for each process)
- M_{opt}: optimal number of sweeps per iteration (min. total number of sweeps for convergence)



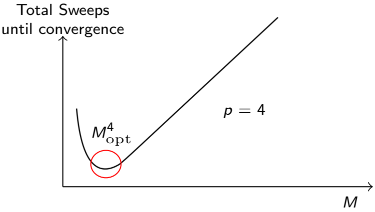
How to judge the speedup?

When changing the number of processes, the parallel implementation yields different simulations.

p : number of processes

M : number of sweeps per iteration (for each process)

M_{opt}: optimal number of sweeps per iteration (min. total number of sweeps for convergence)



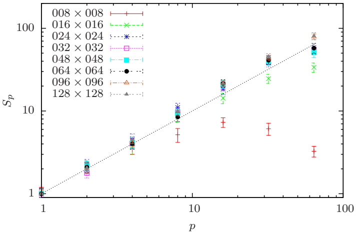
Ising - Speedup

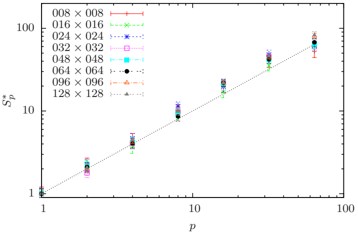
speedup in time

$$S_p = \frac{\bar{t}_1}{\bar{t}_p}$$

speedup in sweeps per core

$$S_p^* = \frac{[\bar{N}_{\text{iter}} M_{\text{opt}}(L,1)]_1}{[\bar{N}_{\text{iter}} M_{\text{opt}}(L,p)]_p}$$

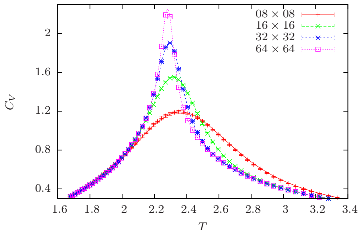


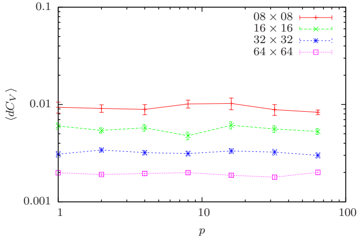


We can see a linear speedup in both definitions.
This means, that with double amount of processes the simulation takes half the time for the same total statistics.

Ising - Precision

The parallel production runs (same total statistics) were analyzed and the average deviations from the exact results were compared.

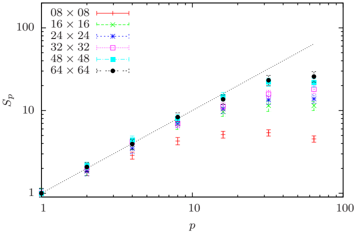


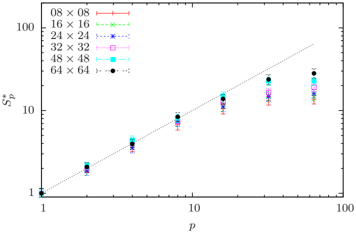


The mean deviation from the exact results remains of the same order for different degrees of parallelization.

8-state Potts - Speedup

A drop in performance may be observed for systems accompanied by barriers like the Potts model in the energy and the Ising model in the magnetization.



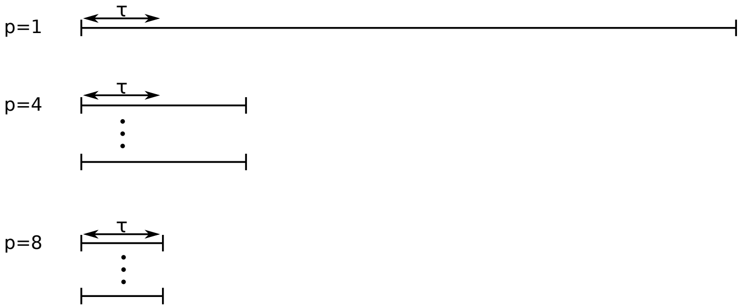


$$S_p = \frac{\bar{t}_1}{\bar{t}_p}$$

$$S_p^* = \frac{[\bar{N}_{\mathrm{iter}} M_{\mathrm{opt}}(L,1)]_1}{[\bar{N}_{\mathrm{iter}} M_{\mathrm{opt}}(L,p)]_p}$$

Natural Limit

The effect can be understood comparing the integrated autocorrelation time τ to the number of sweeps per iteration.



Summary and Outlook

- The simple parallel implementation speeds up the simulation in the weight iteration <u>and</u> the production run.
- A limit is given by possibly emerging barriers and following large integrated autocorrelation times.

- Outlook
  - Combination with advanced variations of MUCA
  - Application to polymer and other continuum models

Acknowledgments

Thank you for your attention

Funding:
European Union and the Free State of Saxony

Supported by:
BuildMoNa, DFH-UFA

BuildMoNa

Gefördert aus Mitteln der Europäischen Union

Europa fördert Sachsen.
ESF
Europäischer Sozialfonds